Conceptual Modeling of Information Systems

Antoni Olivé

Conceptual Modeling of Information Systems

With 185 Figures

Springer

Author

Antoni Olivé

Universitat Politècnica de Catalunya
Department of Software (LSI)
Jordi Girona 1-3,
E-08034 Barcelona
Catalonia, Spain
olive@lsi.upc.edu

Library of Congress Control Number: 2007932289

ACM Computing Classification (1998): D.2.1, D.2.2, H.2.1, I.6.5, F.3.1

ISBN 978-3-540-39389-4 Springer Berlin Heidelberg New York

Typeset by the author
Production: LE-TEX Jelonek, Schmidt & Vöckler GbR, Leipzig
Cover design: KünkelLopka Werbeagentur, Heidelberg

Printed on acid-free paper 45/3180/YL - 5 4 3 2 1 0

Foreword

It is now more than fifty years since the first paper on formal specifications of an information system was published by Young and Kent. Even if the term "conceptual model" was not used at that time, the basic intention of the abstract specification was to a large extent the same as for developing conceptual models today: to arrive at a precise, abstract, and hardware independent model of the informational and time characteristics of a data processing problem. The abstract notation should enable the analyst to organize the problem around any piece of hardware. In other words, the purpose of an abstract specification was for it to be used as an invariant basis for designing different alternative implementations, perhaps even using different hardware components.

Research and practice of abstract modeling of information systems has since the late fifties progressed through many milestones and achievements. In the sixties, pioneering work was carried out by the CODASYL Development committee who in 1962 presented the "Information Algebra". At about the same time Börje Langefors published his elementary message and e-file approach to specification of information systems. The next decade, the seventies, was characterized by the introduction of a large number of new types of, as they were called, "data models". We saw the birth of, for instance, Binary Data Models, Entity Relationship Models, Relational Data Models, Semantic Data Models, and Temporal Deductive Models. At this time, most of the researchers in the modeling field had, essentially, a data-base orientation. I believe the first time the term "conceptual schema" was used was by the ANSI/X3/SPARC, Study Group on Data Base Management Systems, in 1975 when they formulated the "three schema approach" to data-base management. The conceptual schema was seen as the "essential schema", depicting the content of the database in an implementation, and external representation independent way.

The term *conceptual modeling* gradually gained general acceptance, perhaps largely due to the use of the term conceptual schema in the ISO working group's TC97/SC5/WG5 preliminary report, *Concepts and Terminology for the Conceptual Schema* edited by J.J. van Griethuysen, et. al. in 1982. At about the same time information system researchers began to use the term "conceptual modeling" for modeling of information systems in an implementation independent way. Usually, this kind of modeling was carried out during the requirements elicitation and specification phase of systems development. The last two decades of conceptual modeling practice have been dominated by two main trends. The first is the spread and use of the UML object oriented language and approach, including its language OCL (Object Constraint Language) for formulating business rules and constraints. The second trend, in my opinion, is the change of mode of modeling towards a way where users and stakeholders are very much more involved – participatory modeling. This trend points to the importance of modeling skills and knowledge becoming important not only for system development professionals but also for stakeholders and users.

Antoni Olivé has written an impressive book that brings and puts together knowledge of conceptual (and data-) modeling, produced in research spanning more than half a century. In doing so it summarizes, and puts in context, research on conceptual modeling presented in more than 220 references. It deals with all essential aspects of conceptual modeling, thoroughly explained and illustrated in detail. Structural as well as behavioral conceptual modeling concepts are explained in detail. Every chapter is concluded with a bibliographical note that gives the research-oriented reader the possibility of further study through references to works on that particular topic. Each chapter also gives students a challenge to test their newly acquired knowledge by solving a number of problems. A fairly large chapter at the end, describing a case study, illustrates the use of modeling constructs presented earlier. Of practical interest are the frequent translations into UML and OCL of the modelling concepts that are introduced. The book concludes with a chapter on "Metamodeling" and a chapter on "Meta-metamodeling" and Metadata Interchange (XMI), a standard that enables the exchange of data about schemas as well as about schema instances. Metamodeling is also an important mechanism for reasoning about conceptual schema languages of different types and for integrating conceptual models with other kinds of models, such as business and enterprise models.

The book is one of the most informative and comprehensive texts on conceptual modeling published to date. It is very appropriate for students of advanced level university courses in information systems, requirements engineering, or in data base design, as well as for qualified practitioners in the field.

Lund in May, 2007

Janis Bubenko
Prof. em., Dr Techn, Dr. Techn. h.c., ACM Fellow
Department of Computer and Systems Science
Royal Inst. of Technology, Stockholm, Sweden

Foreword

Antoni Olivé has taken the time to create a book that is certain to become essential reading for students of conceptual modeling in Information Systems Engineering. Despite the title, this is not just "another book" on conceptual modeling, data bases and information systems. Antoni Olivé has succeeded in creating a text that brings formalization together with the essentials of information systems engineering in a way that encourages the understanding of both.

The most common modeling approaches in Information Systems Engineering may be classified into approaches that are process-oriented, data-oriented, rule-oriented or object-oriented. Antoni Olivé has been a staunch supporter of rule-oriented approaches in his scientific work. It is impressive to see how he has managed to adopt a holistic method in relating his rule-oriented attitude and background to the other three "schools" of thought.

His objective of explaining in detail the use of the standard Object Constraint Language (OCL) of the Universal Modeling Language (UML) was essential in providing an elegant harmonization of the four approaches. To achieve this he starts out with the very simple assumption that an information system has three main functions: a memory to maintain a representation of the state of a domain, an informative function to provide information on the state of the domain, and an activity function to perform actions that change the state of the domain. This initial assumption rests heavily on his own scientific research on rule-oriented approaches to Information Systems Engineering. He has, nevertheless, managed to explain the essentials of UML/OCL within this framework.

Antoni Olivé has placed more emphasis on formalizing the end result of the process of developing an information system than on the stages leading to the detailed end result. The intermediate stages between the initial conception of a system, through its requirement engineering stage and into the finalization stage where the detailed system solution is hammered out, have been given less emphasis. Much of this is well treated in the literature. Most of the known approaches are not based on well-defined specification languages. It may very well be that Antoni Olivé's explanation of the formal basis of UML/OCL will encourage a re-examination of the requirements engineering phase associated with object-oriented approaches to information systems design, as well as a re-examination of process oriented approaches.

We cannot expect in the future to have only one universal modeling language for Information Systems Engineering. There will always be a need for introducing domain-specific language constructs into the information system's specific modeling language, so we will always be challenged by different modeling languages. Hopefully they may in the future be seen as various dialects of a common family of Information Systems Engineering languages. In order to relate the dialects to each other, we need metamodeling languages. Antoni Olivé's text is rounded off with a treatise on meta-modeling, and he thus prepares the reader for the future need of being able to model the modeling languages.

Trondheim May, 2007

Arne Sølvberg
Professor
The Norwegian University of Science and Technology – NTNU
Dean of Faculty of Information Technology,
Mathematics and Electrical Engineering

To those who are happy to see this book,
and to those who would be happy to see it,
if they were still with us.

Preface

If an information system is able to perform useful actions for persons working in a given domain, it is because the system knows something about that domain. The more knowledge it has, the more useful it can be to its users. Without that knowledge, the system is useless.

Most of the knowledge a system has is concrete or particular. It refers to concrete objects and the relationships they have in the domain at some point in time. Given that many systems work in domains with a very high number of objects and relationships, it is hardly surprising that the concrete knowledge they have is very large. Think, for instance, of bank management systems, where it is usual to find a large number of accounts, loans, etc. for which many details must be known (account holders, balances, transactions, etc.).

However, it is not possible to have concrete knowledge about a domain without a prior *general* knowledge about that domain. A bank management system may know the balances of accounts once it knows that there are accounts in the domain, and that accounts always have a balance. Similarly, the system may know the holders of accounts because it knows that accounts do have holders. Concrete knowledge requires prior general knowledge, which is independent of the concrete objects and relationships existing at any point in time.

This general knowledge also includes rules that must be obeyed (for instance, balances may not be negative), definitions that allow new knowledge to be obtained from existing knowledge (for instance, what is understood as the return on investment), and details of the actions that the users want the system to perform when some condition is satisfied (for example, how to calculate the interest earned by savings accounts).

In the information systems field, we use the name *conceptual modeling* for the activity that elicits and describes the general knowledge a particular information system needs to know. The main objective of conceptual modeling is to obtain that description, which is called a *conceptual schema*. Conceptual schemas are written in languages called conceptual modeling languages. Conceptual modeling is an important part of requirements engineering, the first and most important phase in the development of an information system.

The elicitation of the general knowledge required by an information system is a necessary activity. Information systems cannot be designed or programmed without prior elicitation of the knowledge they need to know. This is captured by one of the principles that guide this book, called the *principle of necessity*: “to develop an information system, it is necessary to define its conceptual schema”.

The only option we have is whether or not to explicitly describe that knowledge. That is, whether or not to write the conceptual schema. Sometimes, system development projects choose not to write the conceptual schema, or they do not have the time to do so. In these cases, the general knowledge is in the designers’ heads only. This has many disadvantages. If there are several designers, they must share this knowledge without an explicit description. User participation is hampered. Once the system has been built, it is likely that the general knowledge will be forgotten. The future evolution of the system will require that general knowledge to be rediscovered. The explicit description of the conceptual schema brings many advantages, especially when it is done in a machine-readable language.

Furthermore, many researchers have put forward, many times, a vision in which the conceptual schema is the only important description that needs to be created in the development of an information system. According to this vision, the building of information systems is completely automated. The only things to be done are to determine the functions that the information system has to perform and to define its conceptual schema (and, probably, the design and construction of the input/output user interface). The huge potential economic benefit of this vision justifies the research and development efforts currently devoted to it, which are being made mainly in the framework of OMG’s Model Driven Architecture. The progress made in other branches of computer science (especially in the field of databases) makes this vision feasible in the mid-term. On the day when the vision becomes a reality we shall be able to say that “to develop an information system it is necessary and sufficient to define its conceptual schema”.

Objectives

The main objectives of this book are:

1. To describe the principles of conceptual modeling independently of particular methods and languages.
2. To describe these principles in the detail required to correctly apply them in real projects and to be able to assess the methods, languages, and tools that are most suitable in those projects.
3. To describe the formal bases of conceptual schemas. However, in this book, the logical formalization is only sketched and is not pushed too far. The book describes the formal bases with extensive use of intuitive ideas and examples.
4. To describe in detail the use of standard UML/OCL as a particular conceptual modeling language.
5. To provide exercises for readers who want to practice and deepen their knowledge by solving exercises.
6. To give bibliographical references for the concepts presented in the book and for the extensions suggested to readers, including further formalizations.

Audience

The book has two intended audiences:

1. Computer science and information systems students who, after an introduction to information systems, databases, and UML, want to know more about conceptual modeling in their preparation for professional practice.
2. Professionals with some experience in the development of information systems who feel a need to formalize their practical experiences or to update their knowledge, as a way to improve their professional activity.

Some prerequisite knowledge is assumed – and necessary – in order to benefit from the book:

1. Knowledge of the fundamental concepts of the language of first-order logic.
2. Knowledge of fundamental concepts of object technology, such as classes, operations, and inheritance.

3. Knowledge of the fundamental constructs of ER and UML for information modeling. A basic knowledge of OCL is necessary from Chap. 8 onwards.

Structure of the Book

The 18 chapters of this book are divided into five logical parts:

- *Chapter* 1: *introduction*. Here we give a general view of conceptual modeling. Readers with prior knowledge about the field may skip this chapter, but it may be useful to those who want to recall concepts and terms learnt long ago.
- *Chapters* 2–10: *structural modeling*. Here we study the concepts of entity types, relationship types, constraints, derivation rules and taxonomies.
- *Chapters* 11–15: *behavioral modeling*. Here we describe the concepts of events, their constraints, and their effects. We also describe behavioral modeling with state machines and statecharts. We include a review of the concept of the use case and its relationship to the conceptual schema.
- *Chapter* 16: *a case study*. In the preceding chapters, we follow a bottom-up approach, starting with the basic elements of entity and relationship types, and then proceeding to more complex elements until we reach state transition diagrams and statecharts. In this chapter, we provide an integrated view of conceptual modeling by means of a case study.
- *Chapters* 17 and 18: *metamodeling*. Here we introduce the main concepts of metamodeling and describe their use. We study two important standards related to metamodeling: the MOF and XMI.

Figure I.1 shows the main precedence relationships among the chapters of this book.

The book also includes a companion website (http://www-pagines.fib.upc.edu/~modeling) where students and professionals can find additional exercises, case studies, reading material and presentations on selected topics. If you have any comments on the book, any typos you have noticed, or any suggestion on how it can be improved, I would like to hear from you. The companion website includes information on how to contact me.

For the convenience of the reader, in this book I use "he" to refer to both genders.

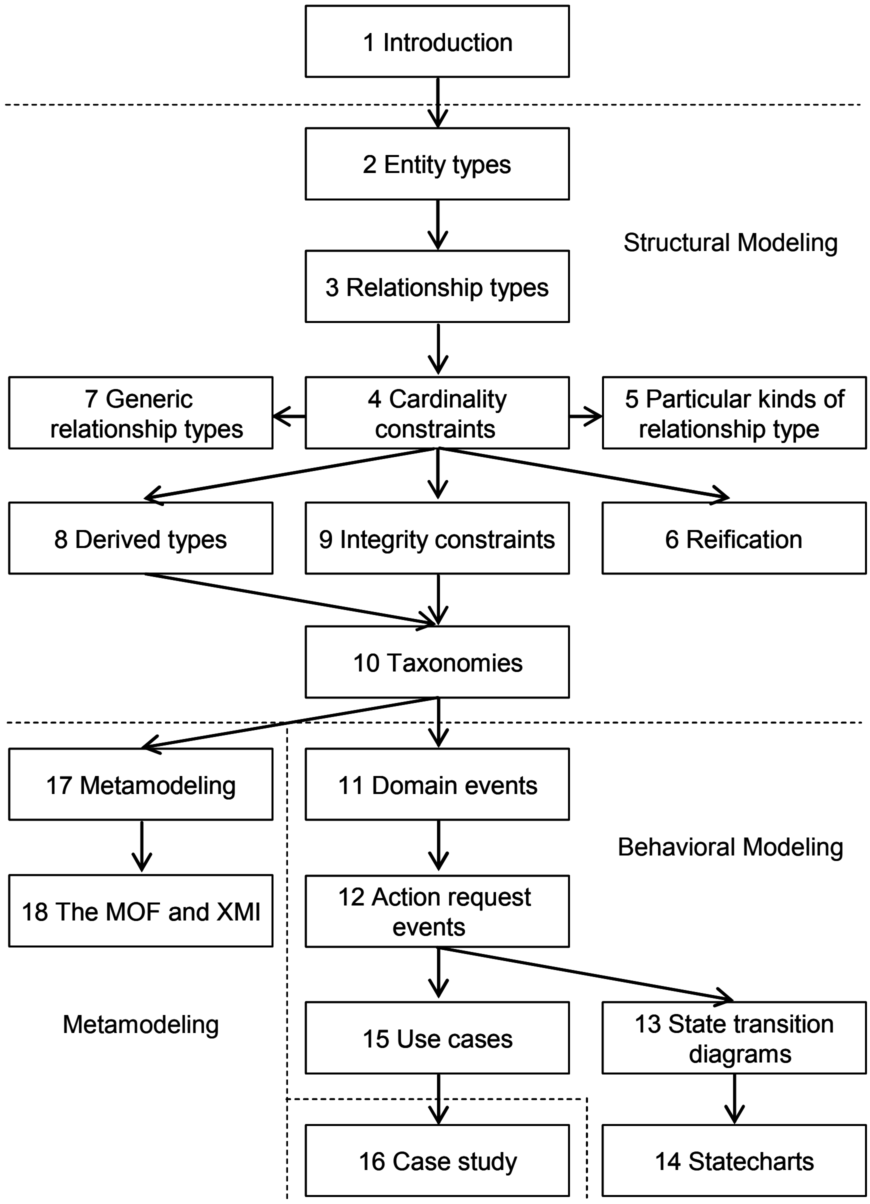

Fig. I.1. Main precedence relationships among the chapters of this book

Acknowledgements

This book explains part of what I have learnt over the years that I have dedicated to the study of conceptual modeling. I have learnt a lot from other people's work. The extensive bibliography shows clearly enough how the ideas and realizations of others have contributed to this book. My first thanks to all these people. I also express my gratitude to the members and friends of the IFIP WG8.1, CAiSE, and ER communities for the many opportunities they have given to me to share and discuss views on what this book is about.

Many people contributed directly and indirectly to bringing this text into existence. First and foremost, I would like to thank my colleagues of the GMC group (whose current composition is: Jordi Cabot, Jordi Conesa, Dolors Costal, Cristina Gómez, Enric Mayol, Joan Antoni Pastor, Anna Queralt, Ruth Raventós, Maria Ribera Sancho, and Ernest Teniente) for providing a friendly environment in which I have been able to discuss many ideas that have helped me to sharpen and focus this book, and for the many comments and suggestions they have given to me over the years.

During the thirteen-year period in which the ideas expressed in this book were developed, from their first expression as lecture notes in Catalan until their current form and language, many friends, colleagues and students have reviewed, read and commented on previous drafts of this book or its chapters. Their help is gratefully acknowledged. Among them, I specially thank Carles Farré, Oscar Díaz, Juan Ramón López, Elena Planas, Carme Quer, Sudha Ram, Keng Siau, Albert Tort and Toni Urpí. Preliminary versions of this book have been used in several courses on conceptual modeling. I deeply appreciate the many comments made by the students. Any remaining errors are, of course, mine.

I would like to thank my university, the Technical University of Catalonia, for the continued support that it has provided to me during these years, and the research agencies that have funded part of my research. Finally, I would also like to thank Ralf Gerstner, from Springer, for his continued interest and help in this book.

Barcelona, June 2007 Antoni Olivé

Contents

1 Introduction 1
1.1 Functions of an Information System 1
1.1.1 The Memory Function 3
1.1.2 The Informative Function 4
1.1.3 The Active Function 6
1.1.4 Examples of Information Systems 7
1.2 Conceptual Modeling 9
1.2.1 The Structural Schema 10
1.2.2 The Information Base 14
1.2.3 The Behavioral Schema 15
1.2.4 Integrity Constraints 18
1.2.5 Derivation Rules 20
1.2.6 The Principle of Necessity for Conceptual Schemas 21
1.3 The Abstract Architecture of an Information System 22
1.4 Requirements Engineering 27
1.5 Quality of Conceptual Schemas 28
1.6 A Brief History of Conceptual Modeling 31
1.6.1 Logical Models 31
1.6.2 Semantic Data Models 32
1.6.3 Conceptual Models of Information Systems 33
1.6.4 Object Orientation 34
1.7 Bibliographical Notes 34

2 Entity Types 37
2.1 Introduction 37
2.1.1 Definitional Concepts 38
2.1.2 Functions of a Concept 39
2.1.3 Prototypical Concepts 39

2.1.4 Exemplar-Based Concepts..40
2.2 Design of Concepts..40
2.3 Definition of Entity Types..41
2.3.1 Names..44
2.3.2 Population..44
2.3.3 Subsumption..45
2.4 Representation in an Information System..45
2.4.1 State of the Information Base..47
2.4.2 Logical Representation..48
2.4.3 Representation in UML..49
2.4.4 Conceptual Models: Single or Multiple Classification..50
2.4.5 Conceptual Models: Static or Dynamic Classification..51
2.4.6 Properties of the Representation..51
2.5 Data Types..54
2.5.1 Data Types in UML..56
2.6 Bibliographical Notes..56
2.7 Exercises..58

3 Relationship Types..59
3.1 Definition..59
3.1.1 Degree..63
3.1.2 Pattern Sentence..63
3.1.3 Unary Relationship Types..66
3.1.4 Population..66
3.1.5 Subsumption..68
3.2 Representation in an Information System..68
3.2.1 State of the Information Base..69
3.2.2 Logical Representation..69
3.2.3 Representation in UML..70
3.2.4 Properties of the Representation..74
3.3 Attributes..75
3.3.1 Conceptual Models Based on Attributes..76
3.3.2 Attribute Pattern Sentence..77
3.3.3 Representation in UML..77
3.3.4 On the Use of Attributes..78
3.4 Bibliographical Notes..80
3.5 Exercises..81

4 Cardinality Constraints..83
4.1 Cardinality Constraints of Binary Relationship Types..83
4.1.1 Existence Dependency Relationship Types..86
4.1.2 Attributes..86

4.1.2 Recursive Relationship Types 87
4.1.3 Satisfiability of Cardinality Constraints 88
4.2 Cardinality Constraints of *n*-ary Relationship Types 90
4.2.1 Consistency and Inference Rules 93
4.3 Maximal Participation 95
4.4 Bibliographical Notes 98
4.5 Exercises 99

5 Particular Kinds of Relationship Type 103
5.1 Reference Relationship Types 103
5.1.1 Simple Reference 104
5.1.2 Compound Reference 105
5.1.3 Set Reference 106
5.2 Identification 107
5.2.1 Identifiability of Entity Types 109
5.3 Replacing Entities with Identifiers in Relationships 109
5.4 Elementary Relationship Types 111
5.5 Decomposing NonElementary Relationship Types 113
5.5.1 Decomposition Based on Functional Dependencies 114
5.5.2 Decomposition Based on Multivalued Dependencies 116
5.5.3 Decomposition by Absorbing a Constant Entity Type 119
5.6 Bibliographical Notes 120
5.7 Exercises 121

6 Reification 123
6.1 Definition 123
6.2 Representation in UML 126
6.2.1 Association Classes 126
6.2.2 Implicit Reification 127
6.2.3 Implicit Reification as a Schema Transformation 129
6.3 Partial Reification 130
6.4 Bibliographical Notes 133
6.5 Exercises 134

7 Generic Relationship Types 137
7.1 Definition 137
7.2 Representation in an Information System 139
7.2.1 Logical Representation 139
7.2.2 Representation in UML 140
7.3 Part–Whole Relationships 141
7.3.1 Description 141
7.3.2 Representation in UML 142

7.3.3 Part Sharing 143
7.3.4 Part Dependency 144
7.4 Grouping 145
7.4.1 Description 145
7.4.2 Representation in UML 147
7.4.3 Homogeneous Versus Heterogeneous Groups 147
7.5 Roles 147
7.5.1 Description 147
7.5.2 Representation in UML 149
7.5.3 Propagation 150
7.6 Materialization 151
7.6.1 Description 151
7.6.2 Representation in UML 152
7.6.3 Inheritance 153
7.7 Bibliographical Notes 153
7.8 Exercises 155

8 Derived Types 157
8.1 Derivability 157
8.1.1 Base Types 157
8.1.2 Derived Types 158
8.1.3 Hybrid Types 159
8.1.4 Transformation of Hybrid Types into Derived Types 159
8.1.5 Design of Derivability 160
8.2 Representation in an Information System 161
8.2.1 Logical Representation 161
8.2.2 Representation in UML 162
8.2.3 Representation of Derivation Rules by Operations 164
8.3 Particular Kinds of Derived Type 167
8.3.1 Derived by Union 167
8.3.2 Derived by Specialization 169
8.3.3 Derived by Exclusion 170
8.3.4 Derived by Participation 171
8.3.5 Transitive Closure 173
8.4 Derivation Rules for Constant Relationship Types 173
8.5 Hybrid Types in UML 175
8.6 Justification for Derived Types 176
8.7 Bibliographical Notes 178
8.8 Exercises 179

9 Integrity Constraints 181
9.1 The Concept of an Integrity Constraint 181

9.1.1 Integrity = Validity + Completeness 181
9.1.2 Integrity Constraints 182
9.1.3 Violation of Integrity Constraints 183
9.1.4 Violation Response Actions 185
9.2 Classification of Integrity Constraints 185
9.2.1 Classification According to Source 186
9.2.2 Classification According to Scope 187
9.2.3 Classification According to Cause of Violation 188
9.3 Representation in an Information System 189
9.3.1 Logical Representation 189
9.3.2 Representation in UML 192
9.3.3 Representation of Constraints by Operations 193
9.4 Particular Kinds of Static Constraint 196
9.4.1 Key Constraints 196
9.4.2 Reference Constraints 197
9.4.3 Inclusion Constraints 198
9.4.4 Disjunction Constraints 199
9.4.5 Covering Constraints 201
9.4.6 Constraints of Recursive Binary Relationship Types 202
9.4.7 Entity Type Cardinality Constraints 204
9.5 Creation-Time Constraints 204
9.6 Bibliographical Notes 206
9.7 Exercises 208

10 Taxonomies 213
10.1 Specialization 213
10.1.1 The *IsA* Relationship 214
10.1.2 Entity Types Derived by Intersection and Multiple Classification 215
10.1.3 The Entity Type *Entity* 216
10.2 Generalization 216
10.2.1 The *Gens* Relationship 217
10.2.2 Constraints on Generalizations 218
10.2.3 Generalization/Specialization Dimension 220
10.2.4 Explicit Subtypes versus Explicit Dimension Attributes 221
10.2.5 Partitions 222
10.3 The Taxonomy of a Conceptual Schema 222
10.3.1 Valid Type Configurations 223
10.3.2 Taxonomic Constraints and Derivability 224
10.3.3 Partitions and Derivability 227
10.4 Relationship Type Refinement 229
10.4.1 Participant Refinement 229

10.4.2 Particular Kinds of Participant Refinement........................232
10.4.3 Cardinality Constraint Strengthening.................................233
10.4.4 Interaction of *IsA* and Cardinality Constraints 236
10.4.5 Derivation Rule Redefinition...236
10.4.6 Redefining a Base Relationship Type as Derived 238
10.5 Constraint Specialization...239
10.6 Specialization/Generalization of Relationship Types................241
10.6.1 *IsA* and *Gens* Between Relationship Types........................241
10.6.2 Reification and Specialization..242
10.7 Bibliographical Notes...243
10.8 Exercises..244

11 Domain Events 247
11.1 Domain Events as Sets of Structural Events..............................248
11.1.1 Structural Events 248
11.1.2 Domain Events 251
11.2 Representation in an Information System.................................252
11.2.1 Domain Events as Entities...252
11.2.2 Logical Representation..255
11.2.3 UML Representation 256
11.3 Domain Event Constraints..257
11.3.1 Logical Representation..258
11.3.2 UML Representation 258
11.4 Event Effects: The Postcondition Approach..............................259
11.4.1 Logical Representation..261
11.4.2 UML Representation 262
11.4.3 The Frame Problem 265
11.5 Event Effects: The Procedural Approach 266
11.5.1 Logical Representation..267
11.5.2 UML Representation 268
11.6 Consistency with the Structural Schema 271
11.7 Bibliographical Notes..272
11.8 Exercises...274

12 Action Request Events..277
12.1 Actions and Action Request Events 277
12.1.1 Scope of this Chapter...281
12.2 Action Request Event Types 281
12.2.1 Characteristics of Action Request Events...........................283
12.2.2 Constraints of Action Request Events 284
12.3 Effects of Queries 285
12.4 Effects of Action Request Events 289

12.4.1 Effects of Domain Event Notifications....294
12.5 Event Specialization....295
12.6 Generating Conditions....296
12.7 Bibliographical Notes....297
12.8 Exercises....297

13 State Transition Diagrams....299
13.1 Finite State Machines....299
13.1.1 Finite Automata....300
13.1.2 Moore and Mealy Machines....300
13.2 Entities as State Machines....301
13.2.1 Entity Life Cycle....305
13.3 State Transition Diagrams in UML....306
13.3.1 Transitions Triggered by Change and Time Events....310
13.3.2 Unexpected-Event Reception....311
13.3.3 Initial State....311
13.3.4 Final State....312
13.3.5 Junction....312
13.3.6 Choice....314
13.4 From Domain and Action Request Events to Call Events....315
13.4.1 Localization of Event Constraints and Effects....317
13.5 Entity Types with Multiple State Transition Diagrams....321
13.6 Bibliographical Notes....322
13.7 Exercises....323

14 Statecharts....325
14.1 The State Hierarchy....325
14.1.1 Simple Composite States....327
14.1.2 State Configuration and Entity Life Cycle....327
14.1.3 Initial Pseudostate....328
14.1.4 Conflicting Transitions....329
14.2 Parallelism....329
14.2.1 Initial Pseudostate....331
14.2.2 Firing Multiple Transitions....332
14.2.3 Fork....332
14.2.4 Join....333
14.3 Bibliographical Notes....333
14.4 Exercises....334

15 Use Cases....337
15.1 Actors....337
15.2 Use Cases....338

15.2.1 Definition 338
15.2.2 Use Case Actors 339
15.2.3 Use Case Specification 340
15.2.4 Relationships Between Use Cases 343
15.2.5 Use Case Model 345
15.3 Mapping Use Cases to Requests 346
15.3.1 Textual References 347
15.3.2 Creation Dependencies 348
15.3.3 Sequence Diagrams 348
15.4 Bibliographical Notes 350
15.5 Exercises 350

16 Case Study 353
16.1 Main Domain Concepts 354
16.2 Store Configuration 354
16.2.1 Store Data 354
16.2.2 Minimum Values 358
16.3 Store Administration 359
16.3.1 Manufacturers 359
16.3.2 Categories 363
16.3.3 Products 366
16.4 Customers 370
16.5 Online Catalog 373
16.5.1 Shopping Carts 373
16.5.2 Orders 376
16.5.3 Show Previous Orders 380

17 Metamodeling 383
17.1 Meta Entity Types 383
17.1.1 Definition 383
17.1.2 Classification Level 385
17.1.3 *InstanceOf* versus *IsA* 387
17.1.4 Monolevel and Multilevel Information Bases 389
17.1.5 Logical Representation 391
17.1.6 Representation in UML 393
17.2 Powertypes 393
17.3 Class Relationship Types 394
17.4 Meta Relationship Types 397
17.4.1 Definition 397
17.4.2 Logical Representation 398
17.4.3 Representation in UML 399
17.5 Metaschemas 400

17.5.1 Definition 400
17.5.2 Example of a Metaschema 401
17.5.3 Levels of a Meta Information Base 405
17.5.4 The Importance of Metaschemas 405
17.5.5 Conceptual Models versus Metaschemas 406
17.5.6 The UML Metaschema 406
17.6 Stereotypes 408
17.6.1 Definition 408
17.6.2 Stereotypes in the Metaschema 410
17.7 Bibliographical Notes 412
17.8 Exercises 413

18 The MOF and XMI 415
18.1 Meta-Metaschemas 415
18.1.1 Definition 415
18.1.2 The MOF 417
18.2 The MOF as a Conceptual Modeling Language 420
18.2.1 The MOF as an ω-metaschema 421
18.3 XMI 422
18.3.1 XMI Representation of Entities and Relationships 423
18.3.2 XMI Representation of UML Schemas 425
18.4 Bibliographical Notes 428
18.5 Exercises 429

References 431

Index 445